Description and Rationale

The Heliophysics data environment (HPDE) is comprised of several domain specific Virtual Observatories (VxOs). Each VxO focuses its efforts on the types of data and research unique to its domain. However, many pertinent research questions demand resources from multiple domains. As such, it would be beneficial for the HPDE to implement a means for the VxOs, as well as added-value services, to communicate with each other in a standardized manner. This is the goal of the SPASE Query Language (SPASEQL).

SPASEQL is based on the SPASE metadata model. SPASE has become an integral part of the HPDE, and is the basis for most descriptions of data resources. The intent is to carry this familiarity over into developing a standardized means of asking and answering questions. SPASEQL is simply the mechanism for standardizing the messaging between VxOs; it is implementation neutral. The query language does not dictate how queries are to be executed. This means that the VxOs use their domain expertise and existing infrastructure to execute queries. SPASEQL provides a front-end to each VxO that allows them to speak the same language to each other.

The number of VxOs is not overwhelming and the point can be made for learning the unique interfaces of each rather than developing a standardized language. However, this has several drawbacks. The resulting software libraries are dependent on the VxO interfaces remaining static. A change to an interface breaks communications. SPASEQL is a static front end to a VxO. The underlying VxO can change, as mandated by its community, however, the interface other VxOs see remains unchanged.

A second concern is the diversity and complexity of underlying data resources. This is evidenced in the ongoing evolution of SPASE. Thus, for each search capability a VxO implements (granules, spacecraft positions, time periods, etc.), it must have a corresponding means of formatting and packaging the results (the results message structure) for its API. As a result, interacting with VxO APIs scales not with the number of VxOs but rather with the total number of search capabilities. If left to develop independently, similar search capabilities at multiple VxOs will undoubtedly be returned via varying means. Thus, a granule search at one VxO may not be directly interoperable with a granule search at another VxO. This makes the integration of multiple VxO results a formidable challenge.

Rather than each VxO developing independent standards for creating response messages it makes sense to leverage the existing SPASE model. This model already contains agreed upon terminology and definitions that have resulted from years of hard work. SPASE already serves as common parlance for data descriptions. This capability can be extended into a common language for VxOs who are discussing these data descriptions. This makes the usage of the HPDE, as a collective whole, a much easier task.

Costs, Ease of Use, and Benefits

SPASEQL can lead to rapid development and reduced costs. Having a standardized communications language means that a question can be asked the same way from one domain to the next. As a result, software can be reused and queries can be saved and re-executed when needed. This leads to easier integration of VxOs and services. It also allows for the potential chaining of VxOs. That is, if we are speaking the same language, the results of one VxO query can easily be transformed into the constraints of a query to another VxO. Overall, SPASEQL should lead to reduced costs, reuse, and easier integration of the HPDE components.

Implementing SPASEQL does involve some costs. SPASEQL is a front-end that standardizes communications. The underlying VxO must be able to transform SPASEQL queries to execute on its underlying systems. The cost of this effort varies from VxO to VxO and from service to service. It is a function of the complexity and capabilities of the VxO or service. For example, VxOs implementing SQL databases (e.g. VITMO, VHO, and VMO/G) may find it difficult to transform complex SPASEQL queries into meaningful SQL queries. On the other hand, a simple plotting service can only accept very specific SPASEQL queries and may find it trivial to implement. As a result, HDMC discussions have resulted in two modifications to SPASEQL

1. SPASEQL is configurable. It allows an implementer to deploy some components of SPASEQL and ignore others. The result of this is that the VxO has the ability to limit the types of questions it can be asked.

2. SPASEQL will be implemented in levels. The first implementation resulted in a complex but highly configurable language. It was found that several VxOs were interested in configuring a subset of SPASEQL functionality yet the complexity of the language still made it difficult to implement. Rather than have one all encompassing language we found it easier to implement in a "layered" fashion.

    a. SPASEQL Level 0 – a controlled enumerated list of SPASE terms – controls which questions can be asked and does not allow complex queries – Trade Offs: easy to write queries as well as execute them by underlying VxOs, should be acceptable for most types of queries, doesn't allow complex queries, limited to certain SPASE values, will not meet the needs of all VxOs

    b. SPASEQL Level 1 – a superset of Level 0 – does everything Level 0 does but adds the ability for complex queries

    c. SPASEQL Level 2 – a superset of Level 1 – does everything Level 1 does but adds the ability to query over any SPASE term – is not limited to the controlled vocabulary of Levels 0 and 1

By having this configurable and layered approach (each layer is configurable) the costs of implementing SPASEQL should be minimized. As each level is a superset of the preceding one, they are compatible. Level 0 is straightforward, easy to implement, and should satisfy the needs of most VxOs and services. Should a VxO need more functionality they can move to a higher SPASEQL level and augment their implementation with added features, yet still be able to execute queries from lower levels.

Additional costs come in the form of retrofitting existing services and VxO APIs, such as VSO who has a fully functioning API and predates SPASE. While SPASEQL can be used for such sites their owners will need to weigh the time and costs in retrofitting existing infrastructure.

Scope

SPASEQL will not handle all components of the HPDE. There will exist things outside the purview of SPASE, and ultimately SPASEQL. Some functionality will have to be implemented by other means. However, SPASEQL does address the basic set of questions that VxOs and services need to ask each other. As a first order interoperability agent it can go a long way in uniting the capabilities of the HPDE.

Some possible use cases that have emerged from the HDMC are:

VSPO will serve as a global repository of resources. SPASEQL can serve as the means for VSPO to harvest new/modified metadata from the VxOs. In a similar fashion, VSPO can provide a SPASEQL interface by which others can collect and browse the metadata of specific VxOs.

VxO to VxO communication in order to answer user queries that span multiple domains. Each VxO aggregating information about other domains is redundant and wastes resources. SPASEQL will allow VxOs to take advantage of other domain VxOs' expertise

Providing a standard means for VxOs to communicate with added-value services.

Examples

Examples and schema can be found at http://vho.nasa.gov and clicking on the "SPASE-QL" button on the top menu